UNIVERSITI TEKNOLOGI MALAYSIA

PROGRAMMING TECHNIQUE II (SCSJ1023)

SEMESTER 2 2018/2019

GROUP PROJECT

Pharmacy System

ABDULAH AHMAD ALSOLAIMAN (A18CS4013)

TAREQ OTHMAN AL-AHDAL (A18CS0331)

1 SCSR

SECTION 08

Lecturer:

MADAM LIZAWATI MI YUSUF

12rd MAY 2018

The work in this page has been done by: Member's Name *Date:* ????

SECTION A: PROJECT DESCRIPTION

This project is listed under the theme of the health care systems that would provide a benefit to the people in the industry of health care. In our project we targeted the issue of the pharmacy and we want to create a system that would take the role of the pharmacy in a health care institute. Our targeted users are doctors who prescribe medicine for their patients. We aimed to make the system easy to use and that it becomes easy for the doctor to manage the prescription of any medicine that the patient should have. We provided the doctor with some useful functions to assist him in this process, the function will be listed later in this report. The program basically provides an interactive User Interface to navigate through several menu options, every option describes its functionality and prompts will be on the screen to tell the user what to key in consequently.

Synopsis:

Describe the synopsis and general concept of the system you are going to develop. Also, you may want to describe how to use the system. It is highly recommended that you support your description with pictures.

This system is trying to approach the problem from an Object-oriented perspective where we found that there are certain attributes that could be encapsulated together in different classes. This system will be a menu derived system where the doctor will choose among the options provided to him (to be continued)

Objective and Scope:

Describe the objective and scope of your system.

Our objects is to identify problems and try to solve them with our software. In the old days people are were doing almost everything entirely on are solve Takes Time, Not Efficient, Wastes a considerable amount of papers
Hard to find details about patients

We decided to make a system that has a solution for the previous problems.

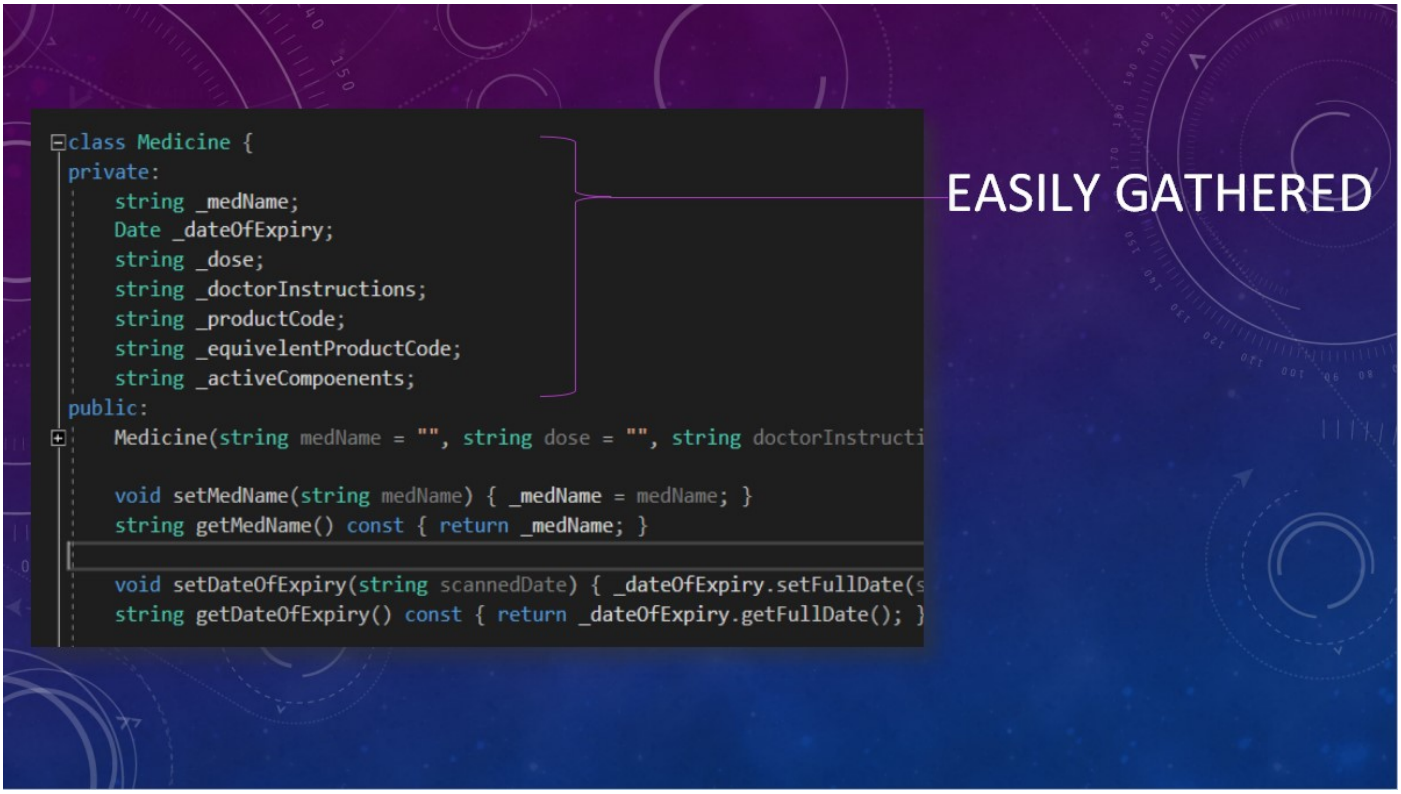
1. Will Make it easy to use
2. Will Save the Doctor's & Patient's Time
3. Will stop using papers as they're no longer needed

Work flow:

Describe how the system works.

OO Concepts:

Encapsulation & Data Hiding:



Encapsulation is basically regarding how data are being gathered in the program, the use of Classes to categorize the data is the encapsulation concept, We used it because it basically makes it easier to handle the functionality wanted by a program through encapsulating attributes and methods into an instance object of a class.

2-Association:

An association is a "using" relationship between two or more objects in which the objects have their own lifetime and there is no owner. ... The objects that are part of the association relationship can be created and destroyed independently.

Under association we have two relations:

2.1-Aggregation:

```
class Doctor : public Person {
private:
    Patient* _listOfPatients[100];
    int _numOfPatients;
public:
    Doctor(string name = "", string id = "") : Person(name, id) {
        _name = name; _id = id; _numOfPatients = 0; _listOfPatients[0] = NULL;
    }


    void setPatient(Patient* p) { _listOfPatients[_numOfPatients] = p; _numOfPatients++; }
    Patient* getPatient(int index) { return _listOfPatients[index]; }
```

The work in this page has been done by: Member's Name *Date:* ????

Why we used aggregation in the class Doctor?

Because aggregation is mainly about the life span of the objects, if the Doctor object get's destroyed, we don't want the person and his all details to be destroyed as well, so we make the aggregation where Dr will point to the patients

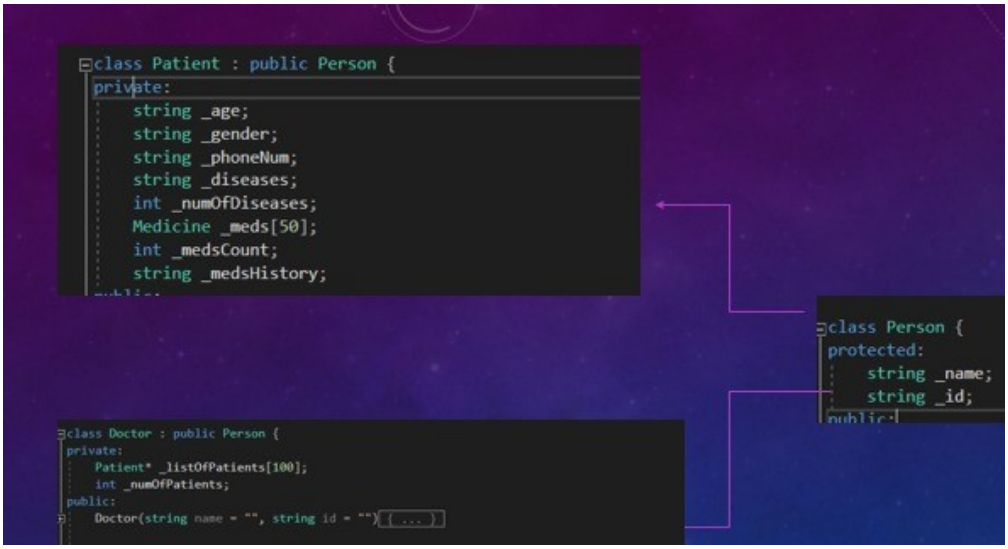
2.2-Composition:

```
class Medicine {
private:
    string _medName;
    Date _dateOfExpiry;
    string _dose;
    string _doctorInstructions;
    string _productCode;
    string _equivelentProductCode;
    string _activeCompoenents;
public:
    Medicine(string medName = "", string dose = "",
```

We used composition because the object Date depends on Medicine, if we delete the medicine, we can't have its expiry date, hence, the composition where date will be destroyed whenever the Medicine is destroyed

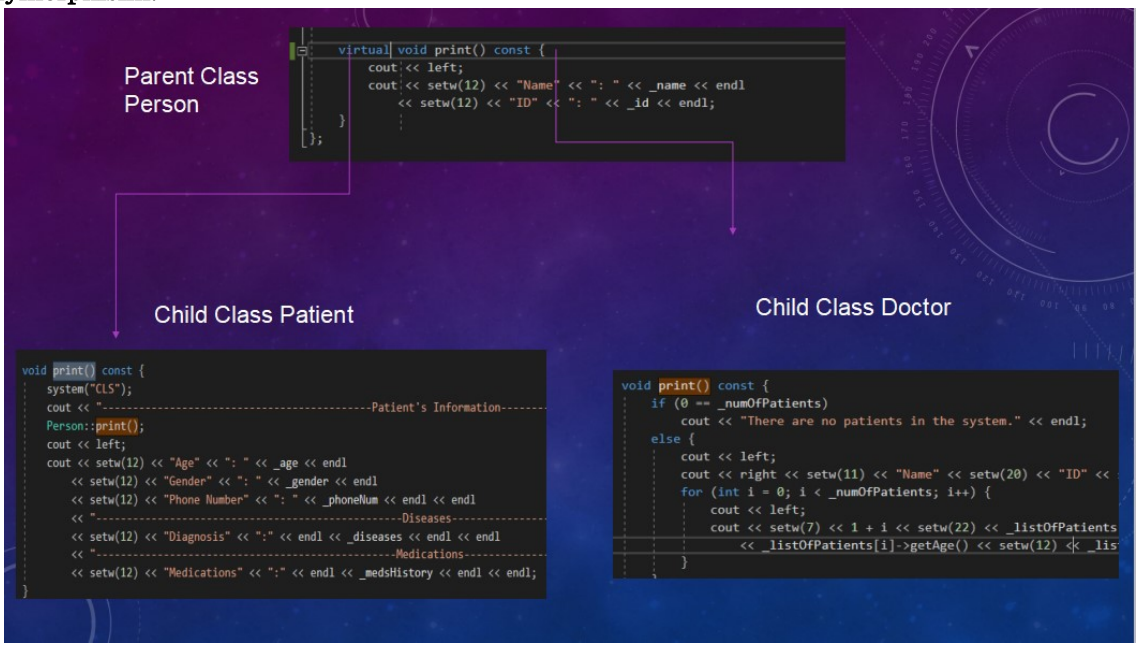
3.0- Inheritance:

Inheritance is a mechanism in which one class acquires the property of another class. For example, a child inherits the traits of his/her parents. With inheritance, we can reuse the fields and methods of the existing class. Hence, inheritance facilitates Re-usability



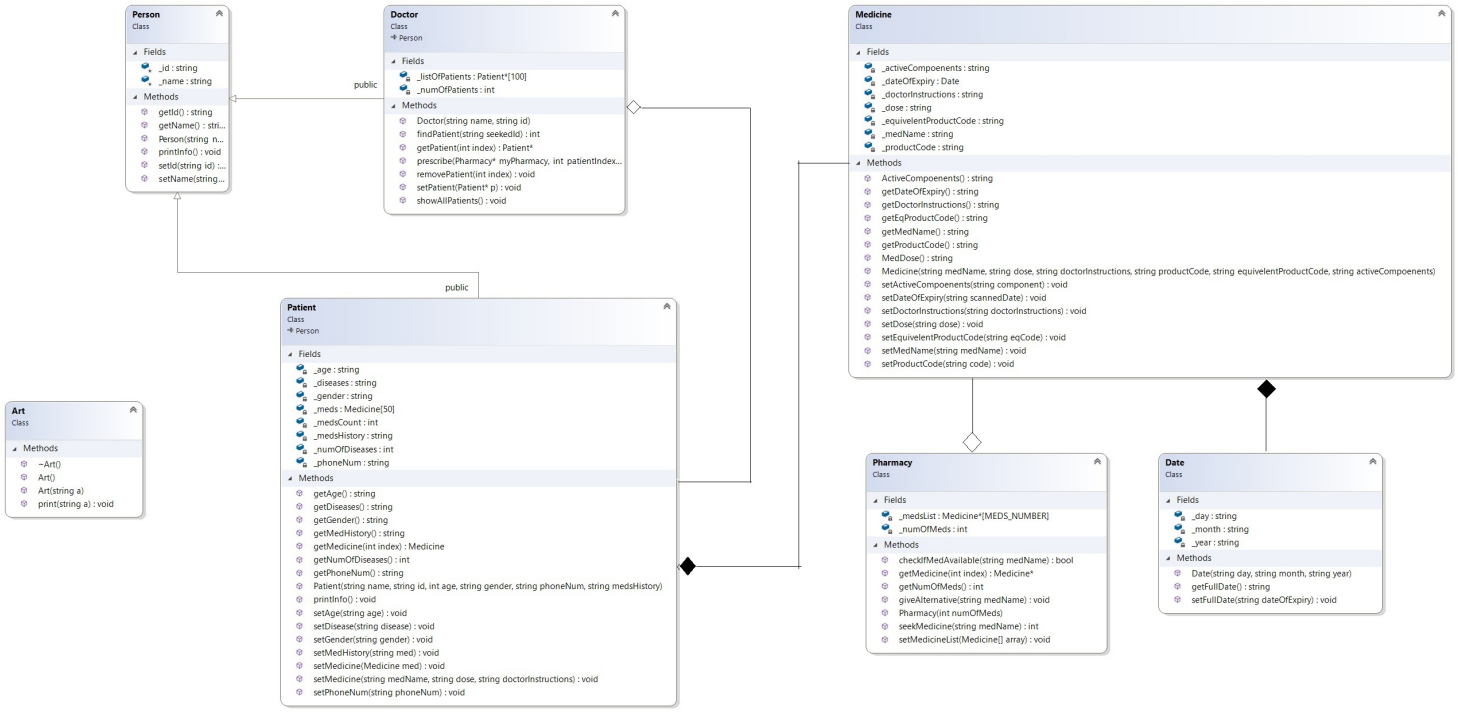
In this program, the appliance of the concept Inheritance was important because it allows us to extend the attributes of the person to reuse them in Class Doctor and Class Patient , yet have at the same time the feature to also give these classes their own special attributes.

4.0- Polymorphisim:



We used that concept here because we would like to call one function from the parent class, yet the information we want to yield from each class is different, hence the polymorphism concept where the same function can act differently according to the class it came from.

SECTION B: CLASS DIAGRAMS



Class Person:

Attributes	Description
_id	Person's ID
_name	Person's name
Methods	**Description**
Person	Class constructor that assign the person's name and ID when the object is created
getName	To get the person's name
setName	To change the person's name
getId	Retrieve the person's ID
setId	To change the person's ID
printInfo	To print the person's information (name and ID)

The work in this page has been done by: Member's Name *Date:* ????

Class Patient:

Attributes	Description
_age	The patient's age
_gender	The patient's gender
_phoneNum	The patient's phone number
_diseases	Diseases that the patient have
_numOfDiseases	Number of diseases that the patient have
_medsCount	Number of medicines that the patient took
_medsHistory	Medicines that user took before (patient's medical record)
_meds	List of Medicines that stores the prescription for the patient
Operations	**Description**
Patient	Class constructor that takes all the necessary data when the the object is created
setAge	Sets the patient's age
getAge	Retrieves the patient's age
setGender	Sets the patient's gender
getGender	Retrieves the patient's gender
setPhoneNum	Sets the patient's phone number
getPhoneNum	Gets the patient's phone number
setMedicine	Sets an object in an array of class medicine
getMedicine	Gets the object from the array of class medicines
Operator>>	Takes the input from the user to fill all the needed information of the patient
setMedHistory	Saves the records of all the medications that the patient took
getMedHistory	Retrieves the records of all the medications that the patient took
setDisease	Sets the disease that the patient has
getNumOfDiseases	Gets the number of diseases that patient has
print	Prints a report-like format that shows the patient's information

Class Doctor:

Attributes	Description
_listOfPatients	Doctor's list of patients
_numOfPatients	number Of Patients
Operations	**Description**
Doctor	Class constructor that takes all the necessary data when the the object is created
setPatient	Assigns a patient
getPatient	Returns the address of the object of class patient
removePatient	Removes the patient from the list
findPatient	Returns the index of the desired patient in the array of patients
prescribe	Prompts the user to key in the patient's diagnosis and the medication that the patient needs
print	prints the information of the list of all patients

Class Date:

Attributes	Description
_day	Days in double digit format
_month	Months in double digits format
_year	Years in double digits format
Operations	**Description**
Date	Constructor that takes all the information (day, month, year)
setFullDate	Extracts full date from a string
getFullDate	Returns full date in dd/mm/yyyy format

Class Medicine:

Attributes	Description
_medName	Medicine name
_dateOfExpiry	Medicine's date of expiry
_dose	Doses of the medicine (how much should the patient take)
_diseases	Diseases that the patient have
_doctorInstructions	Doctor instructions or comments on how to take the medicine
_productCode	Medicine's registered code
_equivalentProductCode	Code for an alternative medicine
_activeComponents	Components of the medicine
Operations	**Description**
Medicine	Class constructor that takes all the necessary data when the the object is created
setMedName	Sets the Medicine's name
getMedName	Retrieves the Medicine's name

setDateOfExpiry	Sets the medicine's date of expiry
getDateOfExpiry	Gets the medicine's date of expiry
setDoctorInstructions	Set the instructions that was written by the doctor
getDoctorInstructions	Retrieves the instructions that was written by the doctor
setProductCode	Assigns a code for the medicine
getProductCode	Gets the product code of the medicine
setEquivalentProductCode	Assigns a code for an alternative medicine
getEqProductCode	Gets the product code of the alternative medicine
setActiveComponents	Sets the components of the medicine
getActiveComponents	Gets the components of the medicine

Class Patient:

Attributes	Description
_age	The patient's age
_gender	The patient's gender
_phoneNum	The patient's phone number
_diseases	Diseases that the patient have
_numOfDiseases	Number of diseases that the patient have
_medsCount	Number of medicines that the patient took
_medsHistory	Medicines that user took before (patient's medical record)
_meds	List of Medicines that stores the prescription for the patient
Operations	**Description**
Patient	Class constructor that takes all the necessary data when the the object is created
setAge	Sets the patient's age
getAge	Retrieves the patient's age
setGender	Sets the patient's gender
getGender	Retrieves the patient's gender
setPhoneNum	Sets the patient's phone number
getPhoneNum	Gets the patient's phone number
setMedicine	Sets an object in an array of class medicine
getMedicine	Gets the object from the array of class medicines
Operator>>	Takes the input from the user to fill all the needed information of the patient
setMedHistory	Saves the records of all the medications that the patient took
getMedHistory	Retrieves the records of all the medications that the patient took
setDisease	Sets the disease that the patient has
getNumOfDiseases	Gets the number of diseases that patient has
print	Prints a report-like format that shows the patient's information

Class Art:

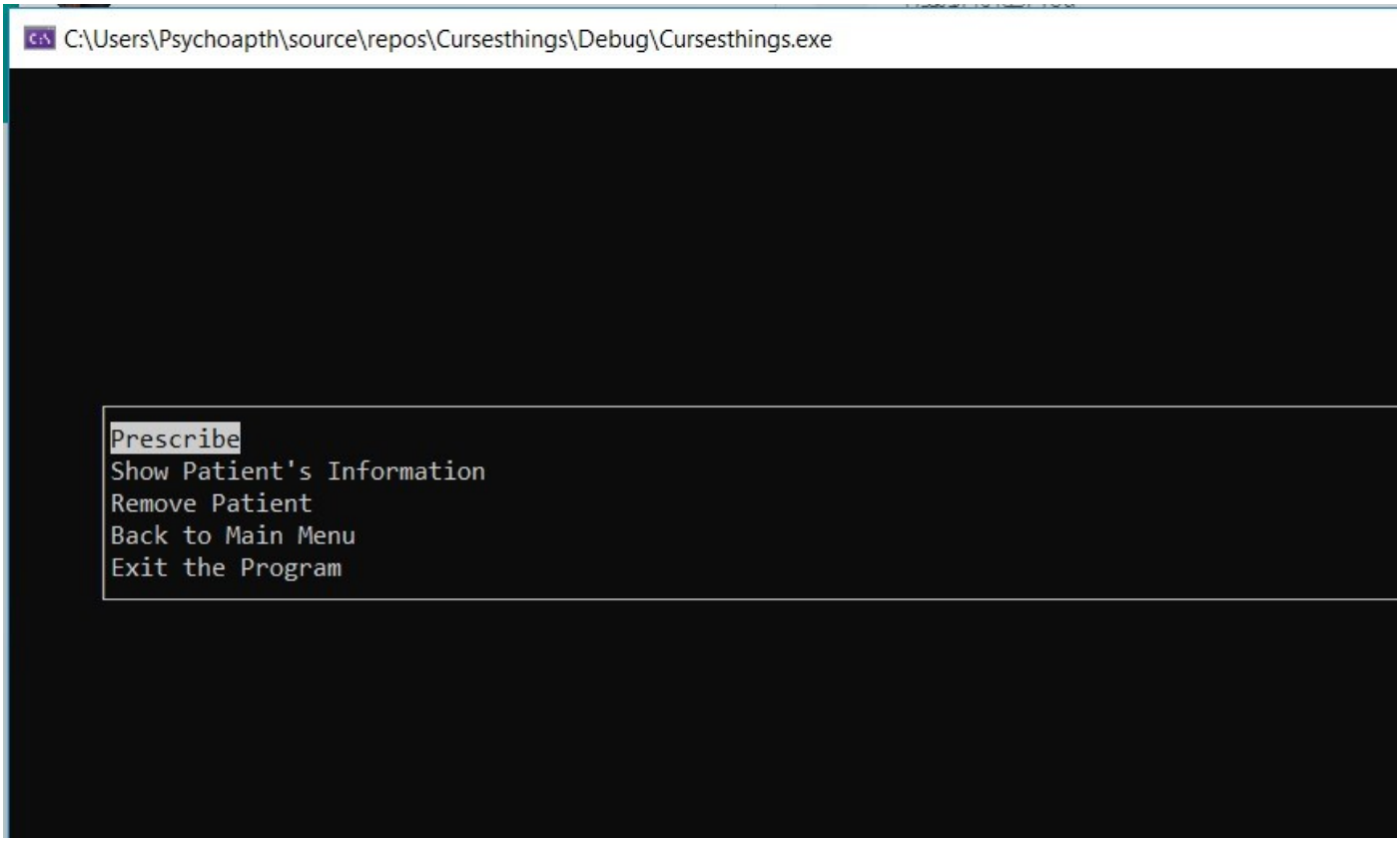
Attributes	Description
Operations	Description
Art	Creates the shapes
~Art	Removes the shapes
print	Prints the output

The work in this page has been done by: Member's Name *Date:* ????

SECTION C: SOURCE CODE AND USER MANUAL

In this section you need to provide the source code of your system. The source code serves as a way to present the object oriented concepts used in your system. You are required to provide the user manual of your system. The user manual describe how to use the system and the flow of your system, i.e show the example of input and the expected output.

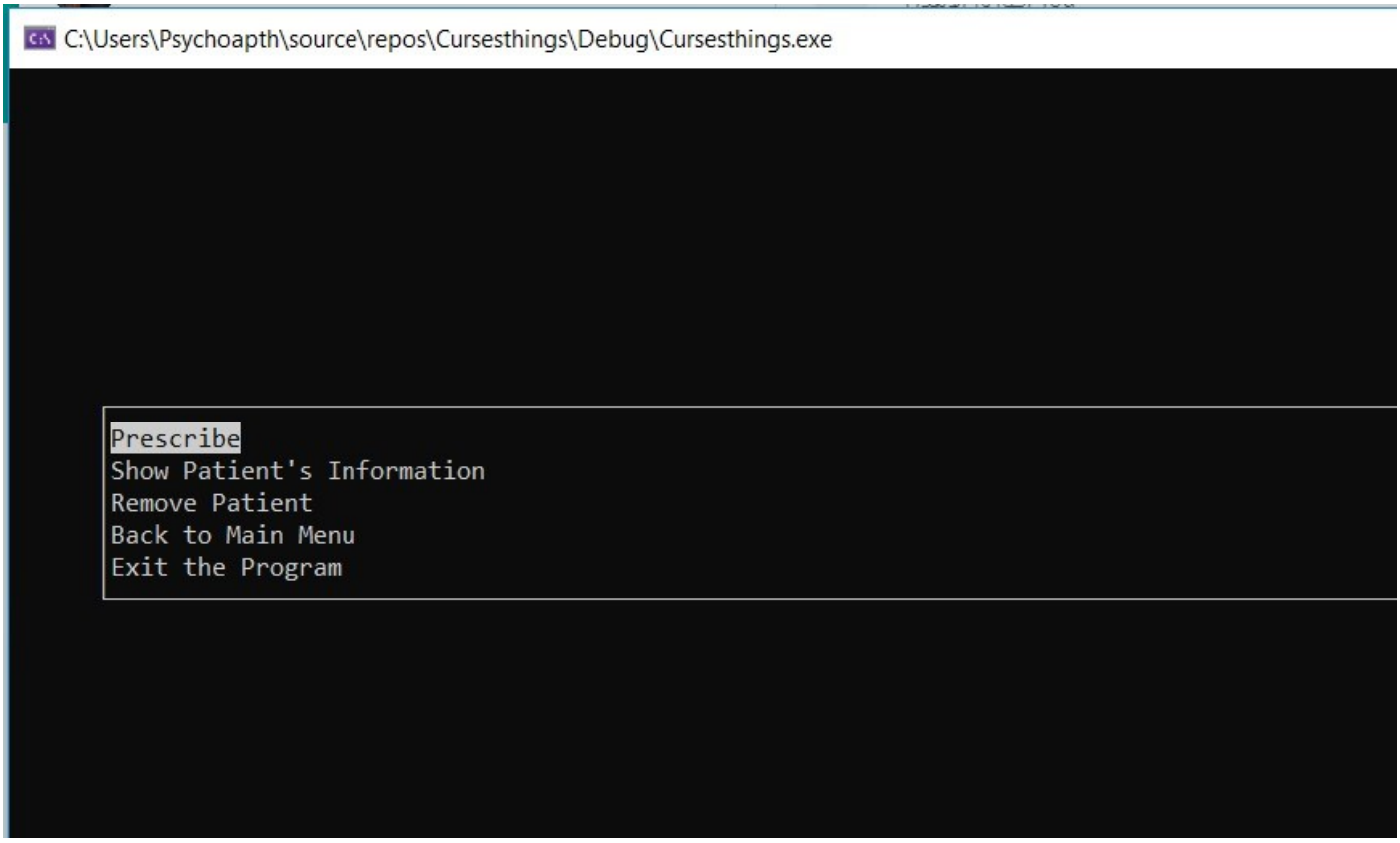
Main Menu:



This screen is the main menu of our application where everything actually starts from. You can see that there are 5 choices available for the user:

1- First one is for the new patient who is not registered in the system. When you click it, you will be prompted to enter the patient's details then you will be redirected to another page where the user needs to provide the diagnosis and the medication that the patient needs.
2- Second one is for the patient who exists in the system. The user will be prompted at first to enter the patient's ID. And Then he will be redirected to a page which asks the user to enter the diagnosis and the medication for the patient. Note the system will display an error message if there are no patients registered.
3- Third option is for displaying every patient that is registered along with there information so that the user (who is the doctor) can know more about his patients such as their names, Id, and phone numbers. Note the system will display an error message if there are no patients registered.
4- Fourth choice is for when the user wants to know something from the pharmacy that he sends his patients to. The user could know more about the medicines and their availability.
5- Fifth choice is for exiting the program.

Patient Menu:

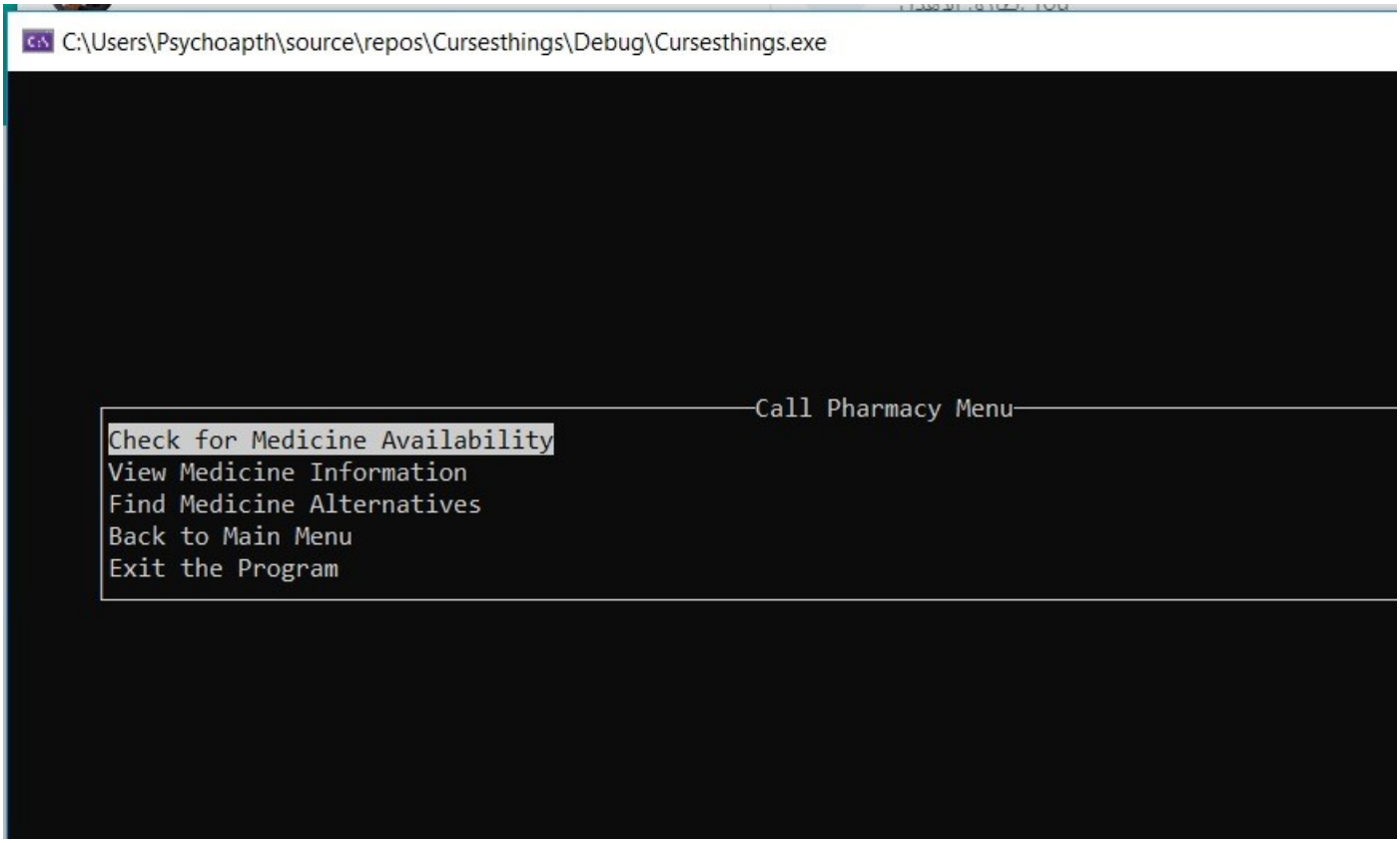


This page is the patient menu where you have multiple choices that you can do for your registered patient. It is redirected from the "existing patient" option (from main menu) and can also be accessed from the "new patient" (from main menu). This page has 5 options

1- The first option is for prescribing the medicine for the patient. This will prompt the doctor to enter his diagnosis and then he enter the number of medicines he is going to prescribe for the patient along his prescription.

2- Second option is to know more about the patent information: The user can check his full name, ID, age, gender, past diagnosis and medications.
3- Third option is for removing the patient information from the system.
4- Fourth option is for going back to the main menu
5- Fifth option is for exiting the application

Pharmacy Menu:



This screen is for the doctor when he wants to know some information from the pharmacy. This page can be accessed from the main menu (option number 4). We can see that this page has 5 options.
1- First option is to check if a certain medicine or not. The user will be redirected to a page where the user
2- Second option is when the doctor wants to know more about a specific medicine which could help him in deciding which medicine he should prescribe to the patient.
3- Third option is for finding an alternative for the medicine if it is not available. The user just need to key in the medicine's name and the alternative will come up if it actually exists
4- Fourth option is for going back to the main menu
5- Fifth option is for exiting the programs